Type-Sensitive Knowledge Base Inference Without Explicit Type Supervision

Prachi Jain*[1] and **Pankaj Kumar***[1] and **Mausam**[1] and **Soumen Chakrabarti**[2]

[1]Indian Institute of Technology, Delhi
{p6.jain, k97.pankaj}@gmail.com, mausam@cse.iitd.ac.in
[2]Indian Institute of Technology, Bombay
soumen@cse.iitb.ac.in

Abstract

State-of-the-art knowledge base completion (KBC) models predict a score for every known or unknown fact via a latent factorization over entity and relation embeddings. We observe that when they fail, they often make entity predictions that are incompatible with the type required by the relation. In response, we enhance each base factorization with two type-compatibility terms between entity-relation pairs, and combine the signals in a novel manner. *Without* explicit supervision from a type catalog, our proposed modification obtains up to 7% MRR gains over base models, and new state-of-the-art results on several datasets. Further analysis reveals that our models better represent the latent types of entities and their embeddings also predict supervised types better than the embeddings learned by baseline models.

1 Introduction

Knowledge bases (KBs) store facts in the form of relations (r) between subject entity (s) and object entity (o), e.g., $\langle \text{Obama}, \text{born-in}, \text{Hawaii} \rangle$. Since KBs are typically incomplete (Bollacker et al., 2008), the task of KB Completion (**KBC**) attempts to infer new tuples from a given KB. Neural approaches to KBC, e.g., Complex (Trouillon et al., 2016) and DistMult (Yang et al., 2015), calculate the score $f(s,r,o)$ of a tuple (s,r,o) via a latent factorization over entity and relation embeddings, and use these scores to predict the validity of an unseen tuple.

A model is evaluated over queries of the form $\langle s^*, r^*, ? \rangle$. It ranks all entities o in the descending order of tuple scores $f(s^*, r^*, o)$, and credit is assigned based on the rank of gold entity o^*. Our preliminary analysis of DistMult (**DM**) and Complex (**CX**) reveals that they make frequent errors by ranking entities that are not compatible with types expected as arguments of r^* high. In **19.5%** of predictions made by DM on FB15K, the top prediction has a type different from what is expected (see Table 1 for illustrative examples).

In response, we propose a modification to base models (DM, Complex) by explicitly modeling type compatibility. Our modified function $f'(s,r,o)$ is the product of three terms: the original tuple score $f(s,r,o)$, subject type-compatibility between r and s, and object type-compatibility between r and o. Our type-sensitive models, **TypeDM** and **TypeComplex**, do not expect any additional type-specific supervision — they induce all embeddings using only the original KB.

Experiments over three datasets show that all typed models outperform base models by significant margins, obtaining new state-of-the-art results in several cases. We perform additional analyses to assess if the learned embeddings indeed capture the type information well. We find that embeddings from typed models can predict known symbolic types better than base models.

Finally, we note that an older model called E (Riedel et al., 2013) can be seen as modeling type compatibilities. Moreover, previous work has explored additive combinations of DM and E (Garcia-Duran et al., 2015b; Toutanova and Chen, 2015). We directly compare against these models and find that, our proposal outperforms both E, DM and their linear combinations.

We contribute open-source implementations[1] of all models and experiments discussed in this paper

*Equal contribution.

[1] https://github.com/dair-iitd/KBI

Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Short Papers), pages 75–80
Melbourne, Australia, July 15 - 20, 2018. ©2018 Association for Computational Linguistics

for further research.

2 Background and Related Work

We are given an incomplete KB with entities $\mathcal{E}$ and relations $\mathcal{R}$. The KB also contains $\mathcal{T} = \{\langle s, r, o\rangle\}$, a set of known valid tuples, each with subject and object entities $s, o \in \mathcal{E}$, and relation $r \in \mathcal{R}$. Our goal is to predict the validity of any tuple not present in $\mathcal{T}$. Popular top performing models for this task are Complex and DM.

In Complex, each entity e (resp., relation r) is represented as a complex vector $\boldsymbol{a}_e \in \mathbb{C}^D$ (resp., $\boldsymbol{b}_r \in \mathbb{C}^D$). Tuple score $f_{\text{CX}}(s, r, o) = \Re\left(\sum_{d=1}^{D} a_{sd} b_{rd} a_{od}^{\star}\right)$, where $\Re(z)$ is real part of z, and $z^{\star}$ is complex conjugate of z. Holographic embeddings (Nickel et al., 2016) are algebraically equivalent to Complex. In DM, each entity e is represented as a vector $\boldsymbol{a}_e \in \mathbb{R}^D$, each relation r as a vector $\boldsymbol{b}_r \in \mathbb{R}^D$, and the tuple score $f_{\text{DM}}(s, r, o) = \langle \boldsymbol{a}_s, \boldsymbol{b}_r, \boldsymbol{a}_o\rangle = \sum_{d=1}^{D} a_{sd} b_{rd} a_{od}$. Earlier, Riedel et al. (2013) proposed a different model called E: relation r is represented by two vectors $\boldsymbol{v}_r, \boldsymbol{w}_r \in \mathbb{R}^D$, and the tuple score $f_{\text{E}}(s, r, o) = \boldsymbol{a}_s \cdot \boldsymbol{v}_r + \boldsymbol{a}_o \cdot \boldsymbol{w}_r$. E may be regarded as a relation prediction model that depends purely on type compatibility checking.

Observe that, in $\langle \boldsymbol{a}_s, \boldsymbol{b}_r, \boldsymbol{a}_o\rangle$, $\boldsymbol{b}_r$ mediates a direct compatibility between s and o for relation r, whereas, in $\boldsymbol{a}_s \cdot \boldsymbol{v}_r + \boldsymbol{a}_o \cdot \boldsymbol{w}_r$, we are scoring how well s can serve as subject and o as object of the relation r. Thus, in the second case, $\boldsymbol{a}_e$ is expected to encode the type(s) of entity e, where, by 'type', we loosely mean "information that helps decide if e can participate in a relation r, as subject or object." Heuristic filtering of the entities that do not match the desired type at test time has been known to improve accuracy (Toutanova et al., 2015; Krompaß et al., 2015). Our typed models formalize this within the embeddings and allow for discovery of latent types without additional data. Krompaß et al. (2015) also use heuristic typing of entities for generating negative samples while training the model. Our experiment finds that this approach is not very competitive against our typed models.

3 TypeDM and TypeComplex

Representation: We start with DM as the base model; the Complex case is identical. The first key modification (see Figure 1) is that each entity e is now represented by *two* vectors: $\boldsymbol{u}_e \in \mathbb{R}^K$ to encode *type* information, and $\boldsymbol{a}_e \in \mathbb{R}^{D'}$ to encode information. Typically, $K \ll D'$. The second, concomitant modification is that each relation r is now associated with *three* vectors: $\boldsymbol{b}_r \in \mathbb{R}^{D'}$ as before, and also $\boldsymbol{v}_r, \boldsymbol{w}_r \in \mathbb{R}^K$. $\boldsymbol{v}_r$ and $\boldsymbol{w}_r$ encode the expected types for subject and object entities.

An ideal way to train type embeddings would be to provide canonical type signatures for each relation and entity. Unfortunately, these aspects of realistic KBs are themselves incomplete (Neelakantan and Chang, 2015; Murty et al., 2018). Our models train all embeddings using $\mathcal{T}$ only and don't rely on any explicit type supervision.

DM uses $(E + R)D$ model weights for a KB with R relations and E entities, whereas TypeDM uses $E(D' + K) + R(D' + 2K)$. To make comparisons fair, we set D' and K so that the total number of model weights (real or complex) are about the same for base and typed models.

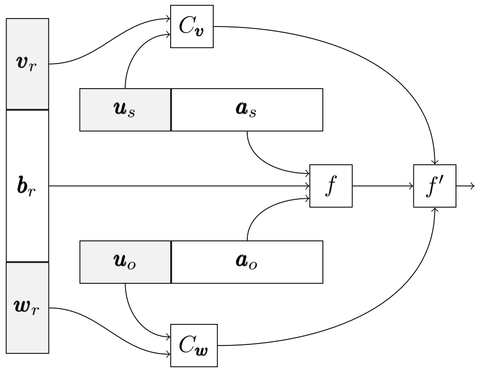

Figure 1: TypeDM and TypeComplex.

Prediction: DM's base prediction score for tuple (s, r, o) is $\langle \boldsymbol{a}_s, \boldsymbol{b}_r, \boldsymbol{a}_o\rangle$. We apply a (sigmoid) nonlinearity:

$$f(s, r, o) = \sigma(\langle \boldsymbol{a}_s, \boldsymbol{b}_r, \boldsymbol{a}_o\rangle), \tag{1}$$

and then combine with two additional terms that measure type compatibility between the subject and the relation, and the object and the relation:

$$f'(s, r, o) = f(s, r, o)\, C_{\boldsymbol{v}}(s, r)\, C_{\boldsymbol{w}}(o, r), \tag{2}$$

where $C_{\boldsymbol{x}}(e, r)$ is a function that measures the compatibility between the type embedding of e for a given argument slot of r:

$$C_{\boldsymbol{x}}(e, r) = \sigma(\boldsymbol{x}_r \cdot \boldsymbol{u}_e) \tag{3}$$

If each of the three terms in Equation 2 is interpreted as a probability, $f'(s, r, o)$ corresponds to a simple logical AND of the three conditions.

We want $f'(s, r, o)$ to be almost 1 for positive instances (tuples known to be in the KG) and close to 0 for negative instances (tuples not in the KG). For a negative instance, one or more of the three terms may be near zero. There is no guidance to the learner on which term to drive down.

Subject s	**Relation** r	**Gold Object** o	**Prediction 1**	**Prediction 2**
Howard Leslie Shore	follows-religion	Jewism (religion)	Walk Hard (film)	21 Jump Street (film)
Spyglass Entertainment	headquarter-located-in	El lay (location)	The Real World (tv)	Contraband (film)
Les Fradkin	born-in-location	New York (location)	Federico Fellini (person)	Louie De palma (person)
Eugene Alden Hackman	studied	Rural Journalism (education)	Loudon Snowden Wainwright III (person)	The Bourne Legacy (film)
Chief Phillips (film)	released-in-region	Yankee land (location)	Akira Isida (person)	Presidential Medal of Freedom (award)

Table 1: Samples of top two DM predictions (having inconsistent types) on FB15K. TypeDM predicts entities of the correct type in top positions in the corresponding examples.

Contrastive Sampling: Training data consist of positive gold tuples (s, r, o) and negative tuples, which are obtained by perturbing each positive tuple by replacing either s or o with a randomly sampled s' or o'. This offers the learning algorithm positive and negative instances. The models are trained such that observed tuples have higher scores than unobserved ones.

Loss Functions: We implement two common loss objectives. The log-likelihood loss first computes the probability of predicting a response o for a query $(s, r, ?)$ as follows:

$$\Pr(o|s,r) = \frac{\exp(\beta f'(s,r,o))}{\sum_{o'} \exp(\beta f'(s,r,o'))} \tag{4}$$

Because $f' \in [0, 1]$ for typed models, we scale it with a hyper-parameter $\beta > 0$ (a form of inverse temperature) to allow $\Pr(o|s, r)$ to take values over the full range $[0, 1]$ in loss minimization.

The sum over o' in the denominator is sampled based on contrastive sampling, so the left hand side is not a formal probability (exactly as in DM). A similar term is added for $\Pr(s|r, o)$. The **log-likelihood loss** minimizes:

$$-\sum_{\langle s,r,o\rangle \in \mathcal{P}} \Big(\log Pr(o|s,r;\theta) \\ + \log Pr(s|o,r;\theta)\Big) \tag{5}$$

The summation is over $\mathcal{P}$ which is the set of all positive facts. Following Trouillon et al. (2016), we also implement the **logistic loss**

$$\sum_{\langle s,r,o\rangle \in \mathcal{T}} \log \Big[1 + e^{-Y_{sro} f'(s,r,o)}\Big] \tag{6}$$

Here Y_{sro} is 1 if the fact (s, r, o) is true and -1 otherwise. Also, $\mathcal{T}$ is the set of all positive facts along with the negative samples. With logistic loss, model weights θ are L2-regularized and gradient norm is clipped at 1.

4 Experiments

Datasets: We evaluate on three standard data sets, FB15K, FB15K-237, and YAGO3-10 (Bordes et al., 2013; Toutanova et al., 2015; Dettmers et al., 2017). We retain the exact train, dev and test folds used in previous works. TypeDM and TypeComplex are competitive on the WN18 data set (Bordes et al., 2013), but we omit those results, as WN18 has 18 very generic relations (e.g., hyponym, hypernym, antonym, meronym), which do not give enough evidence for inducing types.

Model	Embedding dimensions	Number of parameters
E	200	3,528,200
DM+E	100+100	3,393,700
DM	200	3,259,200
TypeDM	180+19	3,268,459
Complex	200	6,518,400
TypeComplex	180+19	6,201,739

Table 2: Sizes were approximately balanced between base and typed models (FB15K).

Metrics: As is common, we regard test instances $(s, r, ?)$ as a task of ranking o, with gold o^* known. We report MRR (Mean Reciprocal Rank) and the fraction of queries where o^* is recalled within rank 1 and rank 10 (HITS). The *filtered* evaluation (Garcia-Duran et al., 2015a) removes valid train or test tuples ranking above (s, r, o^*) for scoring purposes.

Hyperparameters: We run AdaGrad for up to 1000 epochs for all losses, with early stopping on the dev fold to prevent overfitting. All the models generally converge after 300-400 epochs, except TypeDM that exhausts 1000 epochs. E, DM, DM+E and Complex use 200 dimensional vectors. All except E perform best with logistic loss and 20 negative samples (obtained by randomly corrupting s and r) per positive fact. This is determined by doing a hyperparameter search on a set $\{10, 20, 50, 100, 200, 400\}$.

For typed models we first perform hyperparameter search for size of type embeddings (K) such that total entity embedding size remains 200. We get the best results at $K = 20$, from among values in $\{10, 20, 30, 50, 80, 100, 120\}$. This hyperparameter search is done for the TypeDM model (which is faster to train than TypeComplex) on FB15k dataset, and the selected split is used for

	FB15K			FB15K237			YAGO3-10		
Model	MRR	HITS@1	HITS@10	MRR	HITS@1	HITS@10	MRR	HITS@1	HITS@10
E	23.40	17.39	35.29	21.30	14.51	36.38	7.87	6.22	10.00
DM+E	60.84	49.53	79.70	38.15	28.06	58.02	52.48	38.72	**77.40**
DM	67.47	56.52	84.86	37.21	27.43	56.12	55.31	46.80	70.76
TypeDM	75.01	66.07	87.92	38.70	29.30	57.36	58.16	51.36	70.08
Complex	70.50	61.00	86.09	37.58	26.97	55.98	54.86	46.90	69.08
TypeComplex	**75.44**	**66.32**	**88.51**	**38.93**	**29.57**	**57.50**	**58.65**	**51.62**	70.42

Table 3: KBC performance for base, typed, and related formulations. Typed models outperform their base models across all datasets.

all the typed models. To balance total model sizes (Table 2), we choose $K = 19$ dimensions for $\boldsymbol{u}_e, \boldsymbol{v}_r, \boldsymbol{w}_r$ and 180 dimensions for $\boldsymbol{a}_e, \boldsymbol{b}_r$[2].

Typed models and E perform best with 400 negative samples per positive tuple while using log-likelihood loss (robust to a larger number of negative facts as opposed to logistic loss, which falls for class imbalance). FB15K and YAGO3-10 use L2 regularization coefficient of 2.0, and it is 5.0 for FB15K-237. Note that the L2 regularization penalty is applied to only those entities and relations that are a part of that batch update, as proposed by Trouillon et al. (2016). β is set to 20.0 for the typed models, and 1.0 for other models if they use the log-likelihood loss. Entity embeddings are unit normalized at the end of every epoch, for the type models. Also, we find that in TypeDM scaling the embeddings of the base model to unit norm performs better than using L2 regularization.

Results: Table 3 shows that TypeDM and TypeComplex dominate across all data sets. E by itself is understandably weak, and DM+E does not lift it much. Each typed model improves upon the corresponding base model on all measures, underscoring the value of type compatibility scores.[3] To the best of our knowledge, the results of our typed models are competitive with various reported results for models of similar sizes that do not use any additional information, e.g., soft rules (Guo et al., 2018), or textual corpora (Toutanova et al., 2015).

We also compare against the heuristic generation of type-sensitive negative samples (Krompaß et al., 2015). For this experiment, we train a Complex model using this heuristically generated negative set, and use standard evaluation, as in all other models. We find that all the models reported in Table 3 outperform this approach.

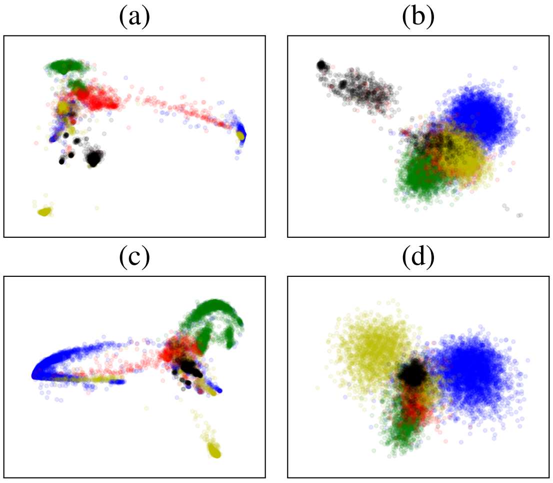

Figure 2: Projection of vectors representing entities belonging to frequent KB types-{**people**, **location**, **organisation**, **film**, **sports**}: a: TypeDM,$\boldsymbol{u}_e$; b: TypeDM,$\boldsymbol{a}_e$; c: TypeComplex,$\boldsymbol{u}_e$; d: DM,$\boldsymbol{a}_e$.

5 Analysis of Typed Embeddings

We perform two further analyses to assess whether the embeddings produced by typed models indeed capture type information better. For these experiments, we try to correlate (and predict) known symbolic types of an entity using the unsupervised embeddings produced by the models. We take a fine catalog of most frequent 90 freebase types over the 14,951 entities in the FB15k dataset (Xie et al., 2016). We exclude `/common/topic` as it occurs with most entities. On an average each entity has 12 associated types.

1. Clustering Entity/Type Embeddings: For this experiment we subselect entities in FB15k that

[2] Notice that a typed model has a slightly higher number of parameters for relation embeddings, because it needs to maintain *two* type embeddings of size K, over and above $\boldsymbol{b}_r$. Using $K = 19$ reduced and brought the total number of parameters closer to that of the base model, for a fair direct comparison. The model performance did not differ by much when using either of the options (i.e., $K = 19$ or 20).

[3] For direct comparisons with published work, we choose 200 and 400 parameters per entity for DM and Complex respectively (Complex model has two 200 dimensional embeddings per entity). DM and TypeDM, on increasing the dimensionality to 400, yield MRR scores of 69.79 and 78.91, respectively, for FB15K.

Method	Embed-ding	Size	H	C	Type F1
TypeDM	$\boldsymbol{u}_e$	19	66.72	66.29	81.77
TypeDM	$\boldsymbol{a}_e$	180	57.89	59.67	75.96
TypeDM	Both	199	**66.75**	**66.29**	82.57
DM	$\boldsymbol{a}_e$	200	51.40	48.12	81.34
TypeComplex	$\boldsymbol{u}_e$	19	65.90	62.97	82.70
TypeComplex	$\boldsymbol{a}_e$	180x2	50.76	48.57	74.75
TypeComplex	Both	379	66.03	63.09	84.14
Complex	$\boldsymbol{a}_e$	200x2	51.56	47.20	81.58
DM+E	$\boldsymbol{u}_e$	19	0.48	2.05	74.66
DM+E	$\boldsymbol{a}_e$	180	49.62	47.24	82.72
DM+E	Both	199	49.66	47.26	82.68
E	$\boldsymbol{a}_e$	200	39.83	37.62	74.23

Table 4: Interpretation of embeddings wrt supervised types: cluster homogeneity H, completeness C, and type prediction F1 score.

belong to one of the 5 types (people, location, organization, film, and sports) from the freebase dataset. These cover 84.88% of FB15K entities. We plot the FB15K entities e using the PCA projection of $\boldsymbol{u}_e$ and $\boldsymbol{a}_e$ in Figure 2, color-coding their types. We observe that $\boldsymbol{u}_e$ separates the type clusters better than $\boldsymbol{a}_e$, suggesting that $\boldsymbol{u}_e$ vectors indeed collect type information. We also perform k-means clustering of $\boldsymbol{u}_e$ and $\boldsymbol{a}_e$ embeddings of these entities, as available from different models. We report cluster homogeneity and completeness scores (Rosenberg and Hirschberg, 2007) in Table 4. Typed models yield superior clusters.

2. Prediction of Symbolic Types: We train a single-layer network that inputs embeddings from various models and predicts a set of symbolic types from the KB. This tells us the extent to which the embeddings capture KB type information (that was not provided explicitly during training). Table 4 reports average macro F1 score (5-fold cross validation). Embeddings from TypeDM and TypeComplex are generally better predictors than embeddings learned by Complex, DM and E. $\boldsymbol{u}_e \in \mathbb{R}^{19}$ is often better than $\boldsymbol{a}_e \in \mathbb{R}^{180}$ or more, for typed models. DM+E with 199 model weights narrowly beats TypeDM with 19 weights, but recall that it has poorer KBC scores.

6 Conclusion and Future Work

We propose an unsupervised typing gadget, which enhances top-of-the-line base models for KBC (DistMult, Complex) with two type-compatibility functions, one between r and s and another between r and o. Without explicit supervision from any type catalog, our typed variants (with similar number of parameters as base models) substantially outperform base models, obtaining up to 7% MRR improvements and over 10% improvements in the correctness of the top result. To confirm that our models capture type information better, we correlate the embeddings learned without type supervision with existing type catalogs. We find that our embeddings indeed separate and predict types better. In future work, combining type-sensitive embeddings with a focus on less frequent relations (Xie et al., 2017), more frequent entities (Dettmers et al., 2017), or side information such as inference rules (Guo et al., 2018; Jain and Mausam, 2016) or textual corpora (Toutanova et al., 2015) may further increase KBC accuracy. It may also be of interest to integrate the typing approach here with the combinations of tensor and matrix factorization models for KBC (Jain et al., 2018).

Acknowledgements

This work is supported by Google language understanding and knowledge discovery focused research grants, a Bloomberg award, an IBM SUR award, and a Visvesvaraya faculty award by Govt. of India to the third author. It is also supported by a TCS Fellowship to the first author. We thank Microsoft Azure sponsorships, IIT Delhi HPC facility and the support of nVidia Corporation for computational resources.

References

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD Conference*. pages 1247–1250. http://ids.snu.ac.kr/w/images/9/98/sc17.pdf.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *NIPS Conference*. pages 2787–2795. http://papers.nips.cc/paper/5071-translating-embeddings-for-modeling-multi-relational-data.pdf.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2017. Convolutional 2d knowledge graph embeddings. *arXiv preprint arXiv:1707.01476* .

Alberto Garcia-Duran, Antoine Bordes, and Nicolas Usunier. 2015a. Composing relationships with translations. In *EMNLP Conference*. pages 286–290. http://www.aclweb.org/anthology/D15-1034.

Alberto Garcia-Duran, Antoine Bordes, Nicolas Usunier, and Yves Grandvalet. 2015b. Combining two and three-way embeddings models for link prediction in knowledge bases. *arXiv preprint arXiv:1506.00999* https://arxiv.org/pdf/1506.00999.

Shu Guo, Quan Wang, Lihong Wang, Bin Wang, and Li Guo. 2018. Knowledge graph embedding with iterative guidance from soft rules. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*.

Prachi Jain and Mausam. 2016. Knowledge-guided linguistic rewrites for inference rule verification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. pages 86–92.

Prachi Jain, Shikhar Murty, Mausam, and Soumen Chakrabarti. 2018. Mitigating the effect of out-of-vocabulary entity pairs in matrix factorization for kb inference. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-18*.

Denis Krompaß, Stephan Baier, and Volker Tresp. 2015. Type-constrained representation learning in knowledge graphs. In *International Semantic Web Conference*. Springer, pages 640–655.

Shikhar Murty, Patrik Verga, Luke Vilnis, Irena Radovanovic, and Andrew McCallum. 2018. Hierarchical losses and new resources for fine-grained entity typing and linking. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Arvind Neelakantan and Ming-Wei Chang. 2015. Inferring missing entity type instances for knowledge base completion: New dataset and methods. *In NAACL* .

Maximilian Nickel, Lorenzo Rosasco, Tomaso A Poggio, et al. 2016. Holographic embeddings of knowledge graphs. In *AAAI Conference*. pages 1955–1961. https://arxiv.org/abs/1510.04935.

Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M Marlin. 2013. Relation extraction with matrix factorization and universal schemas. In *NAACL Conference*. pages 74–84. http://www.anthology.aclweb.org/N/N13/N13-1008.pdf.

Andrew Rosenberg and Julia Hirschberg. 2007. V-measure: A conditional entropy-based external cluster evaluation measure. In *EMNLP Conference*. http://aclweb.org/anthology/D/D07/D07-1043.pdf.

Kristina Toutanova and Danqi Chen. 2015. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*. pages 57–66. http://www.aclweb.org/anthology/W15-4007.

Kristina Toutanova, Danqi Chen, Patrick Pantel, Hoifung Poon, Pallavi Choudhury, and Michael Gamon. 2015. Representing text for joint embedding of text and knowledge bases. In *EMNLP Conference*. pages 1499–1509. https://www.aclweb.org/anthology/D/D15/D15-1174.pdf.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *ICML*. pages 2071–2080. http://arxiv.org/abs/1606.06357.

Qizhe Xie, Xuezhe Ma, Zihang Dai, and Eduard Hovy. 2017. An interpretable knowledge transfer model for knowledge base completion. *arXiv preprint arXiv:1704.05908* https://arxiv.org/pdf/1704.05908.pdf.

Ruobing Xie, Zhiyuan Liu, and Maosong Sun. 2016. Representation learning of knowledge graphs with hierarchical types. In *IJCAI*. pages 2965–2971.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *ICLR*.